Rapport de projet d'application

Projet : Système d'authentification basé sur la reconnaissance faciale

Ingé 3 Majeur Intelligence Artificielle S1 2022

Elaboré par :

Clément Cronier, Théophile Chêne et Adrien Tirlemont élèves à l'ESME-Sudria Encadrant :

Madjid Maidi

➢ Résumé :

Nous avons, au cours de ces dernières semaines pu commencer à répondre à une problématique précise : l'authentification basée sur la reconnaissance faciale.

Tout au long du début projet, nous sommes passés par différentes phases de réflexion. Nous avons commencé, par effectuer une recherche en ce qui concerne les systèmes préexistants. Nous avons alors compris que pour constituer un système de reconnaissance faciale, il faudrait décomposer notre problématique en deux parties distinctes. Tout d'abord la détection de visage puis ensuite la reconnaissance de ce visage.

Pour ce qui est du Deep learning, il nous est apparu comme une évidence que pour apprendre à un modèle comment reconnaître des visages, il nous faudrait une base de données suffisamment conséquente. Il a donc fallu, que nous trouvions une base de données volumineuse et diversifiée pour répondre au mieux à notre problème. Nous verrons dans une prochaine partie sur quels critères nous nous sommes basés afin d'établir ce choix et quel a finalement été notre choix.

De plus, une fois notre base de données prête, nous avons commencé à réfléchir à un modèle de réseau de neurones qui serait le plus optimal pour la reconnaissance de visage. Il a fallu étudier quels modèles existaient déjà, quels étaient leurs performances, sur quel type de réseaux de neurones nous devrions nous focaliser... Nous verrons également dans une prochaine partie que nous avons suivi deux principaux axes de développement en parallèle afin d'étudier quelle piste aboutirait aux meilleurs résultats.

Nous tenons à remercier notre tuteur M.Maidi qui, tout au long du début de notre projet, nous a aidé à comprendre les problématiques de notre sujet. Il nous a également permis d'aboutir à une première solution fonctionnelle grâce à ses conseils et au temps qu'il a investi afin que nous puissions réussir au mieux le lancement de notre projet.

Table des matières :

I. Introduction

Au cours de notre 5ème année à L'ESME Sudria, un projet a été assigné à chaque groupe d'élèves. Pour ce qui est de notre groupe, le sujet porte sur le Deep Learning, plus précisément l'authentification en utilisant la reconnaissance faciale.

Nous avions, par notre année précédente à l'ESME des notions concernant ce domaine grâce à notre projet de 4ème année basé lui aussi sur le Deep Learning. Il s'agissait finalement de reproduire le fonctionnement du cerveau humain et d'appliquer cette technique à notre modèle. Notre objectif principal, était de mettre en place un algorithme permettant une fois le visage détecté de le classifier.

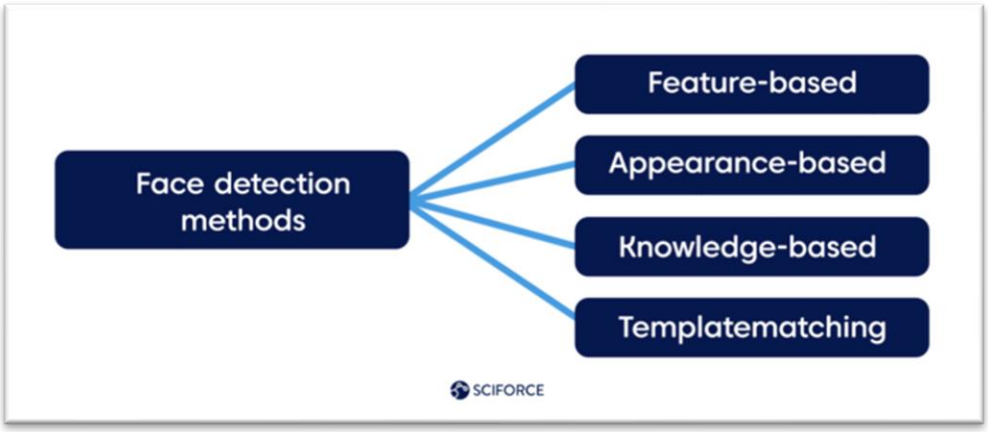
Ainsi, nous avons d'abord imaginé que nous pourrions classifier les visages en se basant sur les points d'intérêts de ceux-ci. C'est au cours d'une discussion avec notre tuteur que nous sommes arrivés à la conclusion que nous devrions classer les visages en se basant sur toute l'image et non sur seulement quelques points d'intérêts. Cependant, afin de mener au mieux notre projet nous continuerons à mener en parallèle la solution basée sur l'extraction des points du visage pour leur classification. Une fois, nos recherches menées à bien, nous avons pu mettre en place un plan d'action de façon à structurer notre projet et le diviser en étapes afin de répondre à la problématique le plus efficacement possible.

Finalement, nous verrons premièrement sur quelles études nous nous sommes basés pour bien comprendre le fonctionnement de la reconnaissance faciale et comment nous avons choisi le dataset qui nous servirait à entraîner notre modèle. Ensuite, nous développerons notre idée en passant en revue chacune des étapes clé de l'autre projet. Puis, nous étudierons la solution que nous avons mise en place afin de répondre à la problématique et à quels résultats nous sommes arrivés. Enfin, nous conclurons en interprétant les résultats auxquels nous avons abouti, puis, afin de pousser notre réflexion plus loin, en étudiant les potentiels axes d'amélioration de notre projet.

II. Revue de la documentation

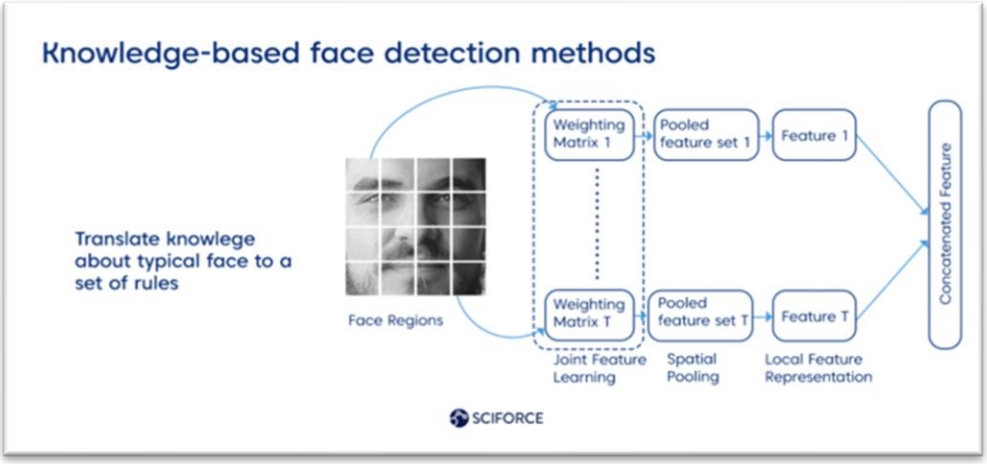
Tout d'abord, afin d'aborder ce projet, il nous a fallu étudier en détails certains concepts afin de commencer le projet de la meilleure façon. Nous avons recherché les différentes méthodes pour la détection de visage.

Etat de l'art de la détection de visage :



Knowledge-based face detection

Cette méthode repose sur l'ensemble des règles élaborées par l'homme selon nos connaissances. Nous savons qu'un visage doit avoir un nez, des yeux et une bouche à certaines distances et positions les uns par rapport aux autres. Le problème avec cette méthode est de construire un ensemble de règles approprié. Si les règles sont trop générales ou trop détaillées, le système se retrouve avec de nombreux faux positifs. Cependant, cela ne fonctionne pas pour toutes les couleurs de peau et dépend des conditions d'éclairage qui peuvent modifier la teinte exacte de la peau d'une personne sur la photo.

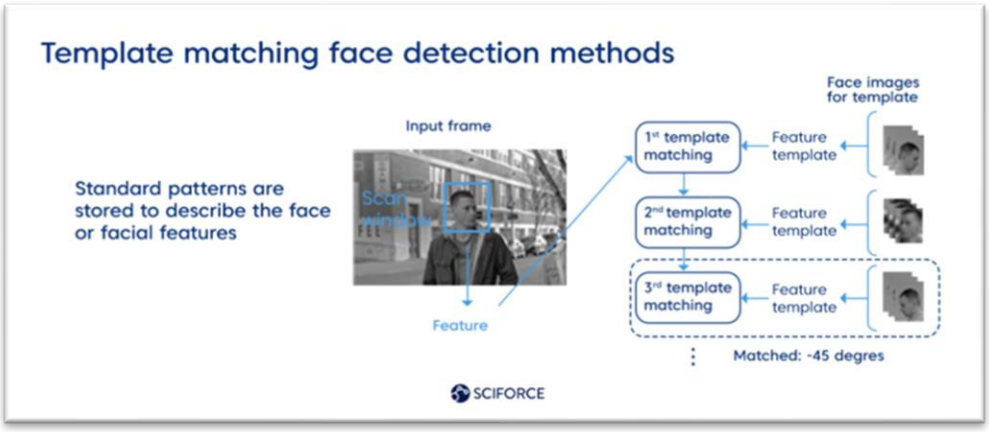


Avantages : facile à comprendre.

Inconvénients : Cela ne fonctionne pas pour toutes les couleurs de peau et dépend des conditions d'éclairage qui peuvent modifier la teinte exacte de la peau d'une personne sur la photo.

Template matching

Le procédé de Template matching utilise des modèles de visage prédéfinis ou paramétrés pour localiser ou détecter les visages par la corrélation entre les modèles prédéfinis ou déformables et les images d'entrée. Le modèle de visage peut être construit par arêtes en utilisant la méthode de détection des arêtes.



Une variante de cette approche est la « controlled background technique ». Si vous avez la chance d'avoir une image de face frontale et un arrière-plan uni, vous pouvez supprimer l'arrière-plan en se basant sur les limites du visage.

Pour cette approche, le logiciel dispose de plusieurs classificateurs pour détecter différents types de visages de face et certains pour les visages de profil, tels que des détecteurs d'yeux, de nez, de bouche et, dans certains cas, même de tout le corps.

Avantages : facile à implémenter.

Inconvénients : C'est de manière générale inapproprié pour la détection de visage.

Feature-based face detection

The feature-based method extrait les caractéristiques structurelles du visage. Il est formé en tant que classificateur, puis utilisé pour différencier les régions faciales et non faciales. Un exemple de cette méthode est la détection de visage basée sur la couleur qui scanne des images ou des vidéos colorées pour les zones avec une couleur de peau typique, puis recherche des segments de visage.

Haar Feature Selection s'appuie sur des propriétés similaires des visages humains pour former des correspondances à partir des traits du visage : emplacement et taille de l'œil, de la bouche, de l'arête du nez et des gradients orientés d'intensités de pixels. Il y a 38 couches de classificateurs en cascade pour obtenir le nombre total de 6061 caractéristiques de chaque face frontale

Avantages : plusieurs modèles pré-entraînés existants, facile à implémenter, bonne accuracy, rapidité d'exécution.

Inconvénients : enclin aux faux positifs/négatifs, régions parfois mal classées ou manquées.

Histogram of Oriented Gradients (HOG) est un extracteur de caractéristiques pour la détection d'objets. Les traits extraits sont la distribution (histogrammes) des directions des gradients (gradients orientés) de l'image.

Gradients ont généralement de grands bords et coins arrondis et nous permettent de détecter ces régions. Au lieu de considérer les intensités de pixel, ils comptent les occurrences de vecteurs de gradient pour représenter la direction de la lumière afin de localiser les segments d'image. La méthode utilise une normalisation de contraste locale superposée pour améliorer la précision.

Avantages : Pas difficile à mettre en place.

Inconvénients : Reconnaît les visages de derrière. Il faudra beaucoup de temps pour l'exécution. Il y aura beaucoup d'informations bruyantes (arrière-plan, flou, éclairs et changements de rotation) que nous ne souhaitons pas considérer comme importantes.

Appearance-based face detection

La méthode « appearance-based » la plus avancée dépend d'un ensemble d'images de visage d'entraînement délégué pour découvrir les modèles de visage. Il s'appuie sur l'apprentissage automatique et l'analyse statistique pour trouver les caractéristiques pertinentes des images de visage et en extraire des caractéristiques.

Avantages : facile à mettre en œuvre, sa robustesse face aux variations de pose, et d'expressions faciales.

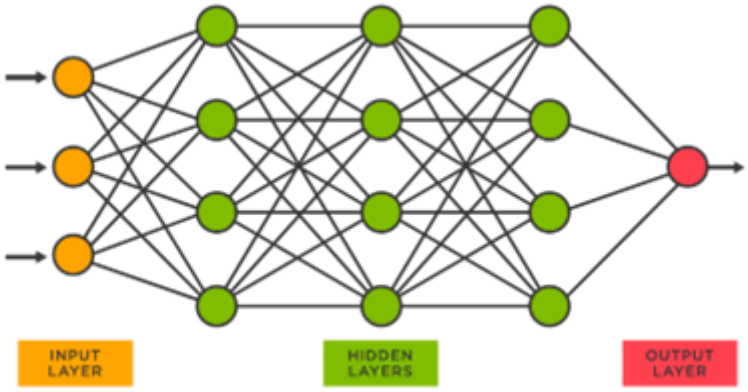
Inconvénients : Couteuse en temps de calcul. Couteuse en espace mémoire. Rend de mauvais résultats quand le nombre d'images d'apprentissage est grand.

Finalement, concernant la détection de visage, nous choisirons de par ses nombreux avantages, la méthode ***Haar Feature Selection***, celle-ci ayant une bonne rapidité d'exécution, une bonne accuracy et possédant de nombreux modèles préexistants.

Ensuite, nous nous sommes intéressés aux meilleures méthodes pour la reconnaissance de visage. Parmi les meilleures méthodes, se trouvent les réseaux de neurones convolutifs (CNN). C'est ce type de réseaux que nous avons décidé d'étudier pour pouvoir mener à bien la reconnaissance des visages détectés.

Les réseaux de neurones convolutifs, appartiennent à une sous-catégorie des réseaux de neurones. Ils sont spécialement conçus pour traiter des images en entrée. Leur architecture est plus spécifique, dans la mesure où elle est composée de deux blocs principaux. Le premier bloc fait la particularité de ce type de réseaux de neurones puisqu'il fonctionne comme un extracteur de caractéristiques. La première couche filtre l'image avec plusieurs noyaux de

convolution. Le résultat de ce premier filtrage est ensuite normalisé et redimensionné. Le second bloc, retrouvé dans tous les réseaux de neurones, est utilisé pour la classification.



Réseau de neurones convolutif

Enfin, concernant le choix de notre dataset, nous avons étudié différents datasets. Dans le cadre de notre projet, il nous fallait des images se rapprochant le plus possible d'un cas d'usage réel. Les images doivent être diversifiées, avoir du contraste, des éclairages différents, des conditions réelles finalement.

Nous avons choisi le Pins Face Dataset, une base de données de photographies de visages conçue pour étudier le problème de la reconnaissance faciale sans contrainte. L'ensemble de données contient entre 100 et 200 images de visages par classe et il est composé de 105 classes. Il s'agit de célébrités. Ce dataset possède un nombre d'images considérable avec justement, des images très variantes et sans contraintes. Cela répondant parfaitement aux problématiques temps réels de notre système.

Nous avons éliminé certains datasets car ceux-ci contenaient des images qui ne correspondaient pas à notre situation de temps réel ou n'étaient pas assez diversifiées par exemple, ou si le nombre d'images était trop faible. Un cas concret de dataset mis de côté a été le Labeled Faces In The Wild dataset. Nous avons après nos études du dataset remarqué que certaines classes ne contenaient qu'une image rendant ainsi le dataset inutilisable. De plus, nous avions utilisé en premier lieu le Kerimelbiler dataset mais celui-ci ne possédant pas assez d'images, notre modèle ne pouvait pas généraliser correctement.

III. Méthodologie de résolution

Comme dit précédemment, pour répondre du mieux possible à notre sujet nous avons dû découper notre problème en de multiples étapes.

En premier lieu, nous renseigner sur le fonctionnement des systèmes de reconnaissance faciale. Ensuite, nous avons réfléchi aux diverses méthodes de détection de visage puis à la reconnaissance du visage. Une fois la bonne compréhension du sujet mise en place, nous nous sommes renseignés sur des potentielles bases de données existantes en discutant avec notre tuteur. Base de données que nous avons finalement pu trouver sur Kaggle. Nous avons passé du temps à la compréhension de la structure de la base de données.

Notre base de données était divisée en deux avec un certain nombre d'images destinées à l'apprentissage de notre modèle et le reste des images destinées à la validation.

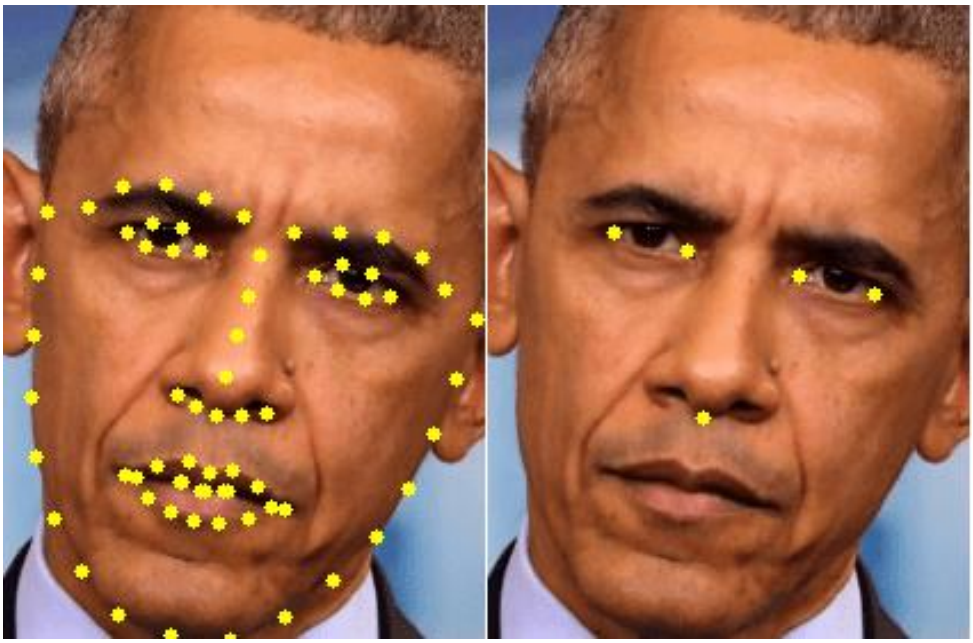
La 2e grande étape a été la mise en place de notre réseau de neurones. En premier lieu nous avons commencé par créer un réseau qui apprendrait à reconnaître les visages en se basant sur les points d'intérêts de ceux-ci. Après échange avec notre tuteur nous avons rapidement vu qu'il serait sûrement plus judicieux de simplement apprendre à notre modèle à reconnaître les visages en lui donnant en entrée toute l'images et non seulement quelques points du visage. Cependant, notre première idée étant intéressante, nous continuerons à explorer cette piste en parallèle de notre réseau de neurones convolutif. Nous utiliserons également un dataset différent dans le cadre de cette approche basée sur les points d'intérêt du visage. Il s'agit du Facial Key Point Detection Dataset qui se compose de 70 000 images. De plus, les images sont très diversifiées et correspondent à des situations en temps réel. L'approche étant différente, nous devions adapter le choix du dataset.

IV. Déploiement de la solution

Pour pouvoir répondre à cette problématique, comme dit précédemment, nous nous sommes renseignés sur le fonctionnement des réseaux de neurones convolutifs. Nous étudierons ici en plus de la méthode de détection de visage, les deux méthodes évoquées précédemment pour la reconnaissance des visages :

1. Méthode de reconnaissance basée sur l'extraction des points d'intérêts du visage :

La méthode de reconnaissance basée sur l'extraction des points d'intérêt se base sur les données biométriques identifiables sur une photo. Ce sont des points d'intérêt comme le centre des iris, la forme du menton, de la bouche etc… Dans cette division du domaine de la reconnaissance faciale, le nombre de marques du visage évolue entre 5 et 68. Evidemment, plus nous avons de marque, plus le système est sécurisé.



Visage avec 68 points *Visage avec 5 points*

Pour pouvoir apprendre à notre modèle à tracer les points, il faut décider de ces données d'entrées, de sortie mais surtout de son type. Ici, nous cherchons à lui faire tracer des points. Ce sont des données réelles, donc nous voulons définir une régression. Les données d'entrées seront les images, et les données de sortie seront les 68 points (donc 136 valeurs, pour les coordonnées (x,y)). Comme dit précédemment, le Dataset choisi, met à disposition 70 000 images pour lesquelles nous avons 68 points d'intérêts.

2. Méthode de reconnaissance basée sur l'image entière :

Dans la reconnaissance d'objet basée sur l'apparence, les caractéristiques sont choisies pour être les valeurs d'intensité de pixel dans une image de l'objet. Ces intensités de pixel correspondent directement à l'éclat de la lumière émise par l'objet le long de certains rayons dans l'espace.

Ainsi, ces caractéristiques sont extraites d'une image du visage et utilisées pour créer une représentation numérique, ou "descripteur de visage", qui capture les caractéristiques uniques de ce visage. Le modèle peut ensuite comparer le descripteur de visage, d'un visage inconnu à une base de données de descripteurs de visages connus et déterminer si le visage appartient à une personne enregistrée dans la base de données.

Les modèles basés sur l'apparence sont généralement formés sur un grand ensemble de données d'images de visages qui ont été étiquetés avec l'identité de la personne dans l'image. Le modèle est alors capable d'apprendre les variations d'apparence faciale associées à différentes personnes. Lorsqu'il est présenté avec un visage inconnu, le modèle utilise ces connaissances apprises pour déterminer à quel visage connu le visage inconnu ressemble le plus.

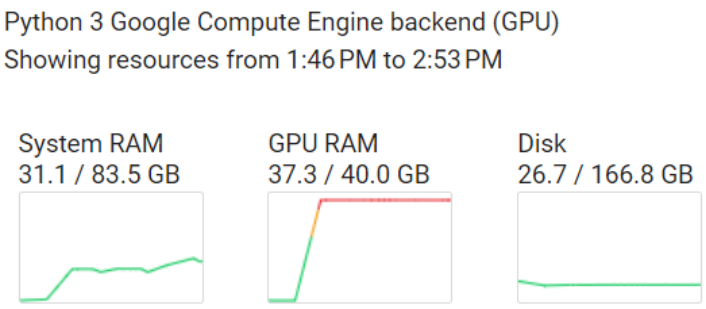
3. Le pré-processing des données :

Tout d'abord, le preprocessing est une étape clé dans la construction de modèles de deep Learning efficaces. Cette étape consiste à préparer les données d'entrée en effectuant des transformations ou des filtrages pour les rendre plus adaptées au modèle. De plus l'importance du preprocessing ne peut être surestimée car il peut grandement affecter la précision et les performances du modèle. Les données brutes peuvent contenir des erreurs, des valeurs manquantes, du bruit ou être mal équilibrées, ce qui peut nuire à la qualité du modèle.

Sur notre Kerimelbiler Data sets, nous n'avons pas effectué de pré-processing, mis à part la normalisation des images, de la réduction de dimension, ainsi que l'encodage des variables étant donné que nous étions assez limitées en puissance de calcul.

Cependant, malgré la réduction de précision des variables et l'optimisation du chargement de données le programme restait très lourd et prenait une quantité importante de RAM, mais soutenable étant donné que nous n'avion que près de 2562 images.

Dans un autre temps, pour la deuxième base de données utilisée, celui-ci étant de près de 17534 images, il nous a fallu faire un important travail d'optimisation lors du pré-processing étant donné que nous utilisions presque toute la GPU RAM de la machine, comme vous pouvez le voir ci-dessous.



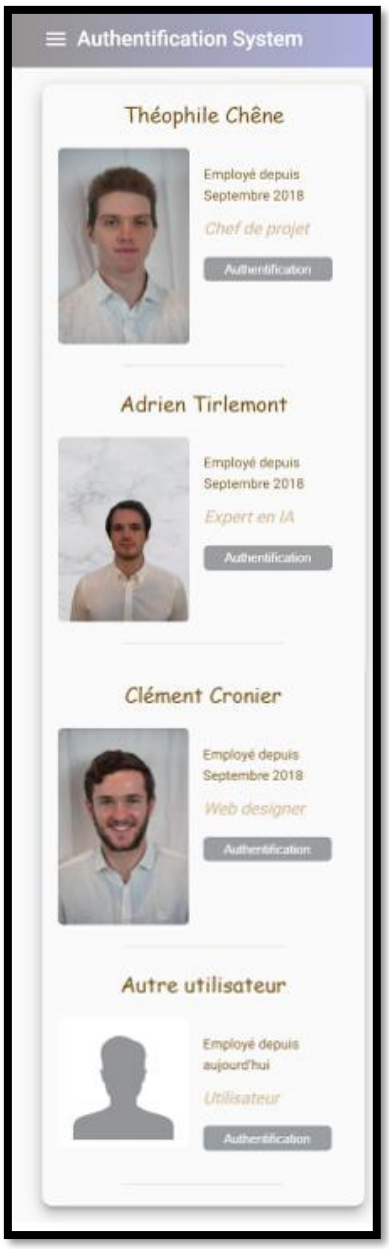
Ainsi, pour palier à ce problème, nous avons essayé plusieurs solutions et avons finalement utilisées plusieurs techniques :

- Normalisation et réduction de la taille des images
- Réduction de la précision des variables
- Conversion des images en niveau de gris
- Conversion des matrices au format tensor (plus optimisé)

Ces modifications, nous ont permis d'obtenir des variables moins lourdes pour entrainer le modèle, et de meilleures qualités.

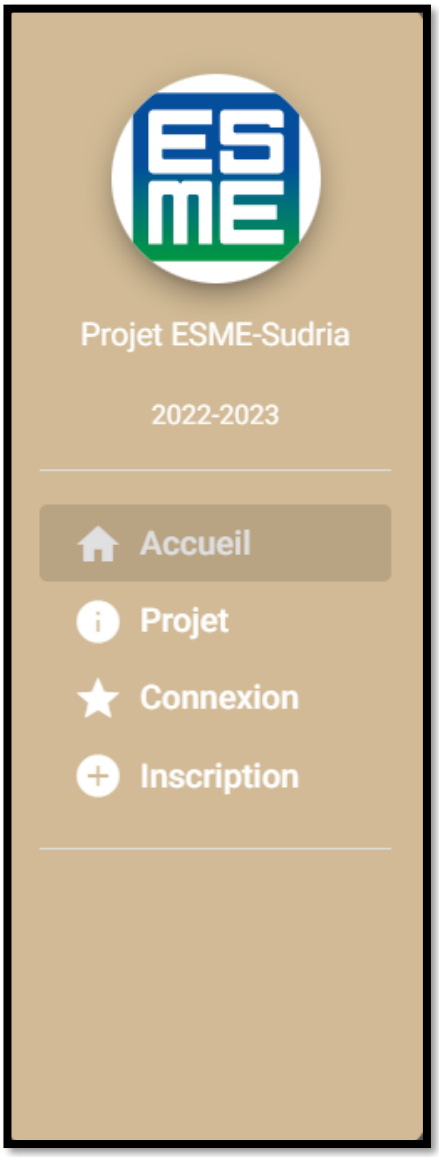
4. Application d'authentification

Bien entendu, dans le cadre de notre projet nous avons mis en place une interface permettant l'application de nos modèles dans un contexte de temps réel. Voici comme se présente notre interface et plus précisément toutes ses composantes.

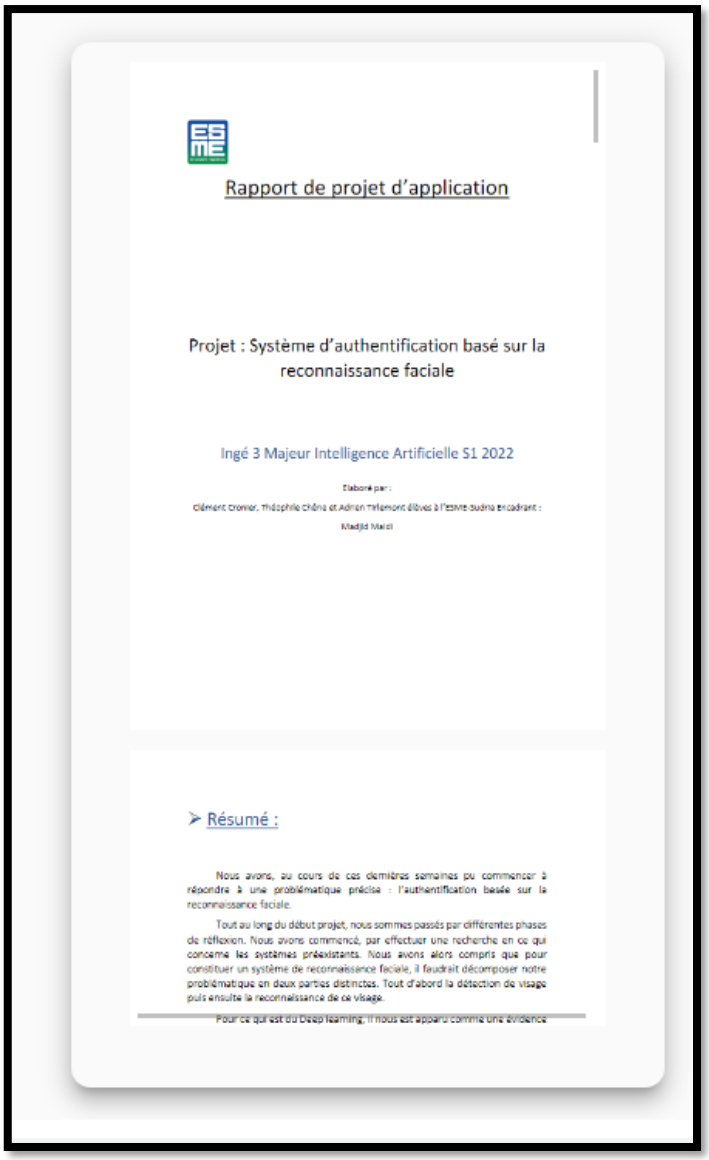


Voici comme se présente la page d'accueil de notre application, on y retrouve nos 3 profils, étant déjà enregistrés dans la base de données. En appuyant sur authentification, cela renvoie l'utilisateur vers l'onglet de connexion.

Un menu déroulant permet de naviguer entre les différents onglets de l'application.



Premièrement, l'onglet « Projet », on y retrouve en format PDF notre rapport de projet.



Deuxièmement, l'onglet « Connexion ». Par défaut, voici ce qui s'affiche à l'écran lorsque l'on n'est pas connecté.

Enfin, l'onglet « Inscription », on y retrouve le mini-formulaire d'inscription afin de pouvoir plus tard identifier la personne lors de sa connexion.

Lors de la bonne authentification d'un individu, voici la page retournée. On y retrouve son nom, son prénom ainsi que l'indice de confidence avec lequel la personne a été reconnue.

Pour le fonctionnement de la connexion, on récupère l'image de la caméra, on fournit cette image aux modèles. Si un modèle estime la précision suffisamment grande selon un indice de confiance, on renvoie le résultat sinon on applique cette image au modèle suivant. Si aucun modèle ne reconnait l'individu, on part du principe que celui-ci n'est pas enregistré et un message d'erreur s'affiche à l'écran.

V. Résultats obtenus et analyse

Nous rappelons que la précision (accuracy en anglais) est une métrique de performance qui évalue la capacité d'un modèle de classification à bien prédire à la fois les individus positifs et les individus négatifs.

De la même manière, la perte (loss en anglais) est une métrique qui évalue l'écart entre les prédictions réalisées par le réseau de neurones et les valeurs réelles des observations utilisées pendant l'apprentissage.

Ces deux métriques nous ont été très utiles pour juger des performances de nos modèles. Nous cherchions à maximiser la précision tout en minimisant la perte.

1. Résultats obtenus pour la méthode de reconnaissance basée sur l'extraction des points d'intérêts du visage :

Cette méthode constituant une régression, nous nous baserons essentiellement sur la perte pour pouvoir évaluer ses performances. Cette perte sera définie par l'erreur quadratique moyenne. Nous avons choisi cet estimateur, car nous voulons pénaliser les grandes erreurs pour pouvoir obtenir des points les plus représentatifs possibles de la réalité. Nous avons cherché à la minimiser au cours de nos différents entrainements. Après nos divers entrainements, nous obtenons une erreur quadratique moyenne de 0.0177. Ceci est une erreur très faible. Nous avons ensuite tracé les points d'intérêt sur un de nos visage pour observer s'il tracer correctement les points sur des données inconnues.

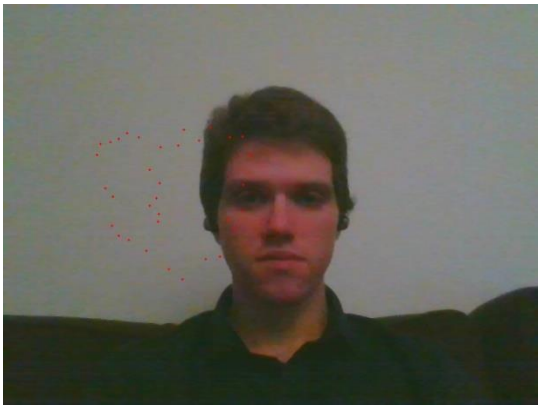


Image inconnu du dataset

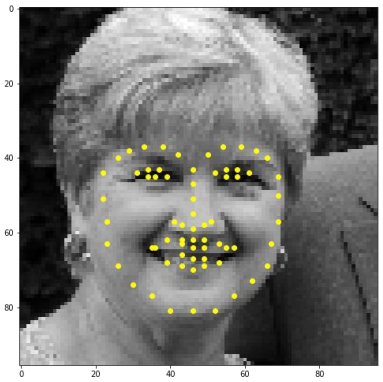
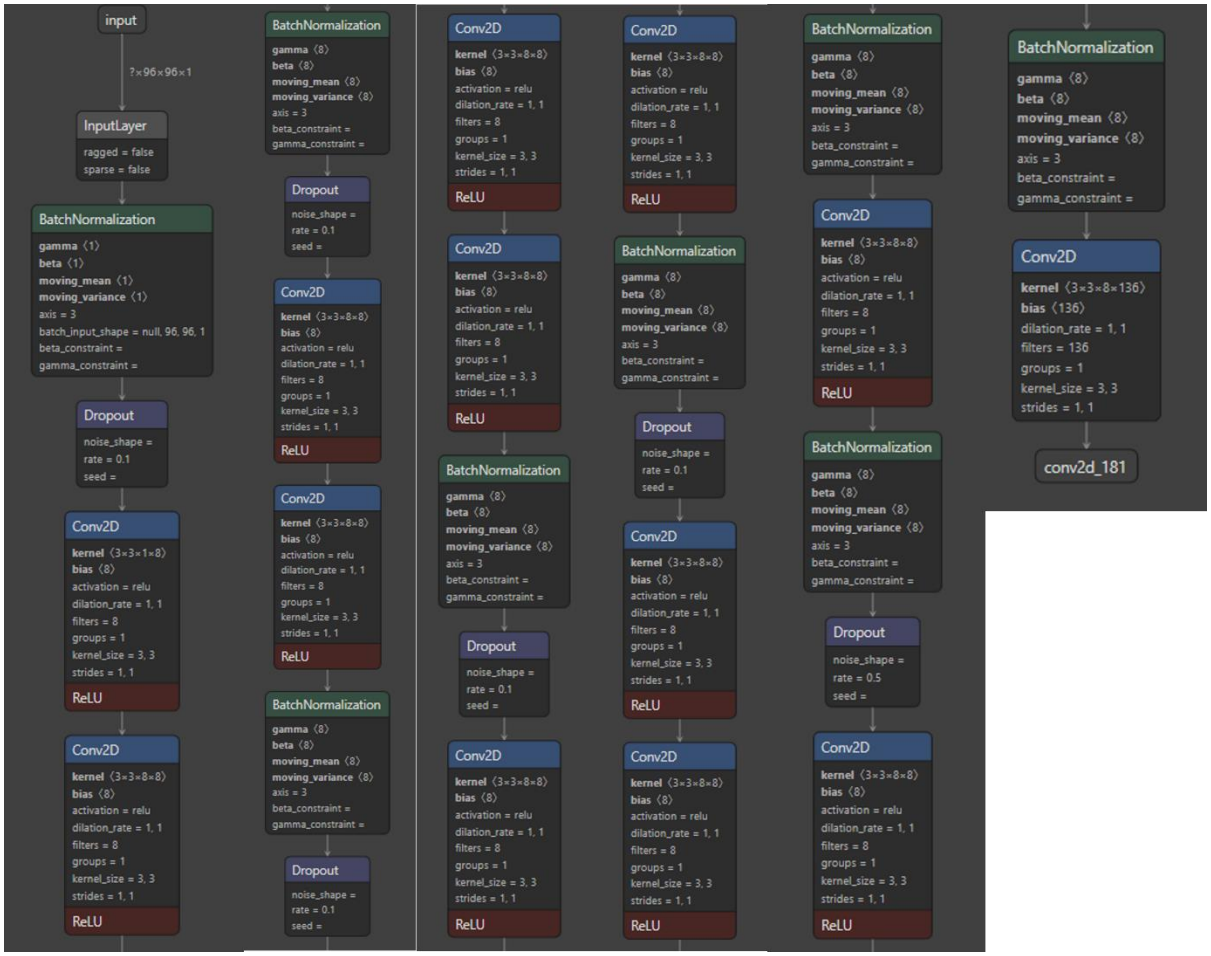


Image de test du Dataset

En continuant l'entrainement de ce réseau nous avons pu diminuer encore plus sa perte sans tomber dans le surentrainement. Elle est maintenant de 0.00144. Cette perte est très faible et nous traçons encore mieux les facial Landmark. Voici les deux architectures des modèles avec leurs paramètres :



En observant les différents tracés du réseau sur des visages étrangers au Dataset, nous avons constaté un décalage des points. En étudiant la manière dont le réseau apprenait, nous avons trouvé la source du décalage. Le décalage vient des landmarks initiaux. En effet, d'une part, les images sont redimensionnées en carré de 96 pixels de large au lieu de carré de 512 pixels, d'autre part, comme nous multiplions les coordonnées pour les afficher sur une image, nous introduisons un deuxième décalage. Enfin, sur le graphique, pour pouvoir tracer des points, les coordonnées doivent être de type entier.

En résumé, nous nous retrouvons avec deux arrondis, l'un issu d'une réduction de la taille des images, l'autre issu du tracé des points sur le graphique. Après avoir déterminé la source du problème, nous avons essayé de contrôler au moins une des sources de ce décalage : les coordonnées d'entrée. Pour cela, nous avons multiplié chaque point par le facteur de réduction de chaque image, soit 96/512. En soumettant les nouvelles données au réseau, celui-ci n'arrivait pas à bien généraliser. La fonction de perte restait au-dessus de 40, malgré les différentes architectures que nous avons essayées.

De plus, nous avons réfléchi à la manière de comparer deux visages avec les Landmarks. La manière la plus intuitive étant de comparer deux mêmes points sur les deux images et de mesurer la distance. Ensuite, nous ferons une moyenne sur les 68 points pour établir la similarité entre les deux visages. Cette similarité sera ensuite comparée à une valeur seuil pour décider de la présence ou non de la personne. Pour déterminer la distance, nous avons recherché les différentes méthodes. Il existe quatre méthodes principalement utilisées pour calculer cette distance.

Entre deux points A(x1, y1) et B(x2, y2) :

5. Distance euclidienne

La distance euclidienne est donnée par la formule :

$$d = \sqrt{((x2 - x1)^2 + (y2 - y1)^2)}$$

6. Distance de Manhattan

La distance de manhattan, aussi appelée distance de taxicab. Cette méthode calcule la distance en prenant la somme des différences absolues des coordonnées entre deux points. Sa formule est donc :

$$d = |x2 - x1| + |y2 - y1|$$

7. Distance de Chebyshev

Cette méthode calcule la distance en prenant la plus grande différence absolue des coordonnées entre deux points. La distance de Chebyshev est donnée par la formule :

$$d = max(|x2 - x1|, |y2 - y1|)$$

Il existe d'autre méthode comme la distance géodésique qui calcule la distance sur une surface incurvé, mais celle-ci n'est pas pertinente pour notre cas.

Chaque méthode possède des avantages et des inconvénients que nous pouvons résumer au tableau suivant :

Distance	Avantage	Inconvénient
Euclidienne	• C'est une méthode simple et facile à comprendre. • Elle est facile à calculer pour des espaces à deux ou trois dimensions.	• Elle ne fonctionne pas bien pour des espaces de dimensions supérieures. • Elle ne prend pas en compte les chemins réels entre deux points, ce qui peut être important dans certaines situations.
Manhattan	• Elle est facile à calculer pour des espaces à deux ou trois dimensions. • Elle est utile pour des situations où les chemins sont restreints à des directions perpendiculaires.	• Elle peut donner des résultats incohérents dans des situations où les chemins ne sont pas limités à des directions perpendiculaires. • Elle peut être moins précise que la distance euclidienne dans certains cas.
Chebyshev	• Elle est facile à calculer pour des espaces à deux ou trois dimensions. • Elle est utile pour des situations où les chemins sont limités à des directions diagonales.	• Elle peut donner des résultats incohérents dans des situations où les chemins ne sont pas limités à des directions diagonales. • Elle peut être moins précise que la distance euclidienne dans certains cas.

Après avoir établi les avantages et inconvénients, nous avons convenu que la distance euclidienne était la plus adaptée à notre cas d'usage, notamment pour sa précision par rapport à la distance de Chebyshev, mais aussi parce que nous n'utilisons que les distances diagonales.

Une fois la méthode établie, nous avons conclu quant au décalage. Ce décalage étant inhérent à chaque tracé, si nous comparons les distances entre chaque point sans aucun traitement en sortie du réseau, les résultats ne seront pas impactés.

2. <u>Résultats obtenus pour la méthode de reconnaissance basée sur l'image entière :</u>

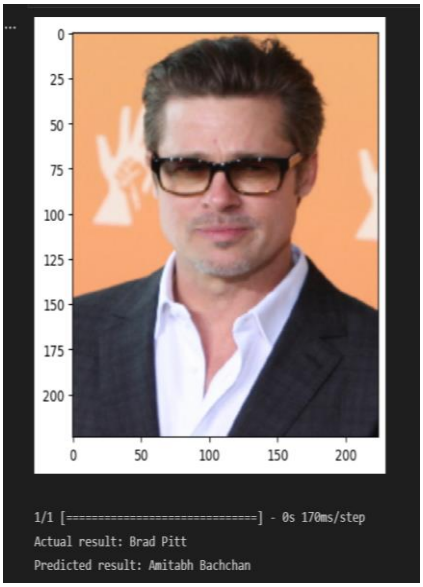


Figure : Example d'image utilisé

Pour la méthode basée sur l'image entière, nous avons choisi deux indicateurs de performance comme la perte et la précision catégorique. Nous obtenons ainsi les résultats suivants, une perte de 0.1608, et une précision sur les données d'entrainement de 0.9514. A la vue de ceux-ci, on peut en déduire qu'il y a une beaucoup de mauvaises prédictions. Nous chercherons à améliorer les prédictions. D'autre part pour celle-ci, le résultat est bon mais on cherchera par la suite à obtenir un résultat se rapprochant de 98-99%, en essayant différents modèles, de manière à obtenir une reconnaissance la plus fiable possible.

Ainsi, après avoir effectué tous ces tests, nous aborderons d'abord les différents modèles en fonction des bases de données utilisées. Puis les techniques afin de réduire le surentrainement. Et finalement, nos résultats après avoir utilisé le transfer learning à partir de modèles existants.

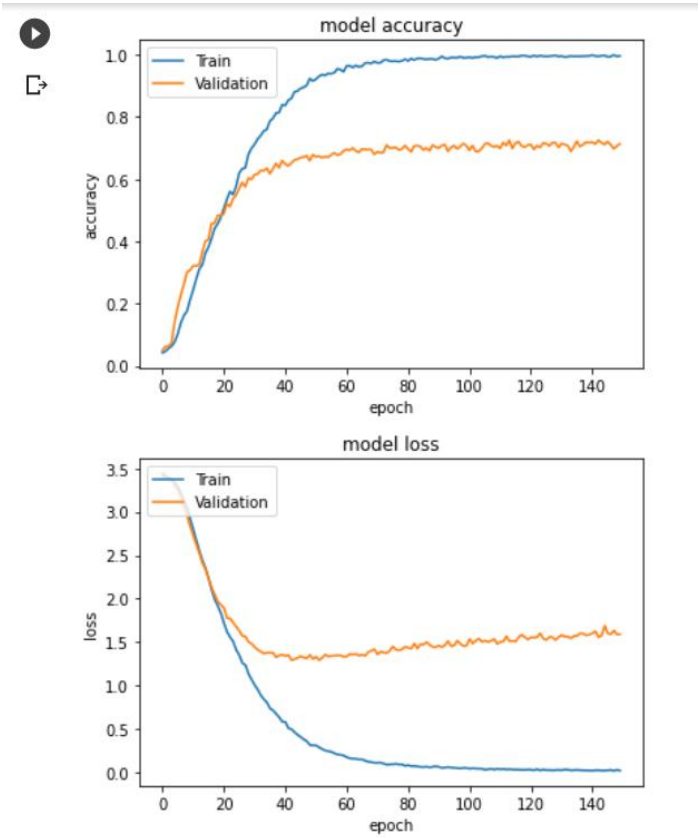
a. Les modèles que nous avons créés :

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 154, 154, 36)      5328

max_pooling2d (MaxPooling2D  (None, 77, 77, 36)       0
)

conv2d_1 (Conv2D)            (None, 73, 73, 54)        48654

max_pooling2d_1 (MaxPooling  (None, 36, 36, 54)       0
2D)

flatten (Flatten)            (None, 69984)             0

dense (Dense)                (None, 1024)              71664640

dropout (Dropout)            (None, 1024)              0

dense_1 (Dense)              (None, 512)               524800

dropout_1 (Dropout)          (None, 512)               0

dense_2 (Dense)              (None, 256)               131328

dropout_2 (Dropout)          (None, 256)               0

dense_3 (Dense)              (None, 31)                7967

=================================================================
Total params: 72,382,717
Trainable params: 72,382,717
Non-trainable params: 0
```

Architecture du modèle entrainé sur la base de donnée Kerimelbiler

Ce premier modèle apprenait mal et s'est révélé trop complexe avec ses 72 millions de paramètres. Il ne nous a pas permis d'obtenir de résultats convenables. La précision maximale obtenue était de 73% sur les données de test, mais de 99% sur les données d'entrainement, comme on peut le voir ci-dessus.

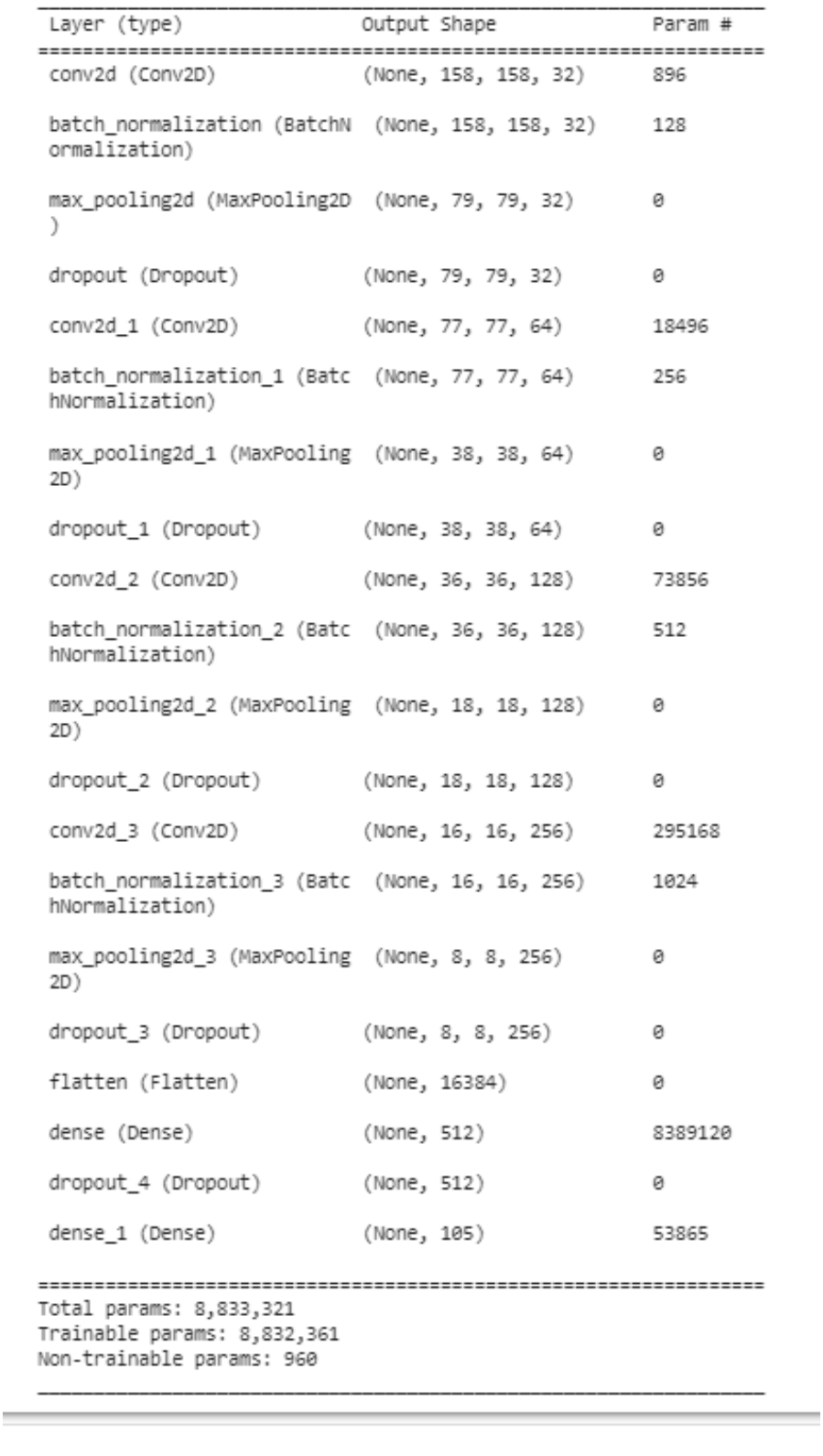
De plus, son architecture est inspirée de modèles préexistants. Nous avons pu observer lors de l'état de l'art, que les couches de conv2d et max_polling2d donnaient de bons résultats sur les premières couches du modèle, et était suivies de couches Dense. Ainsi, nous avons abouti à ce modèle ci, en procédant de façon empirique.



Courbe de précision et perte pour le modèle entrainé

sur la base de donnée Kerimelbiler

Nous sommes ensuite passé sur la deuxième base de données, contenant près de 17 000 images. Le modèle ayant obtenu les meilleurs résultats était celui-ci :



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 158, 158, 32)	896
batch_normalization (BatchNormalization)	(None, 158, 158, 32)	128
max_pooling2d (MaxPooling2D)	(None, 79, 79, 32)	0
dropout (Dropout)	(None, 79, 79, 32)	0
conv2d_1 (Conv2D)	(None, 77, 77, 64)	18496
batch_normalization_1 (BatchNormalization)	(None, 77, 77, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 38, 38, 64)	0
dropout_1 (Dropout)	(None, 38, 38, 64)	0
conv2d_2 (Conv2D)	(None, 36, 36, 128)	73856
batch_normalization_2 (BatchNormalization)	(None, 36, 36, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 18, 18, 128)	0
dropout_2 (Dropout)	(None, 18, 18, 128)	0
conv2d_3 (Conv2D)	(None, 16, 16, 256)	295168
batch_normalization_3 (BatchNormalization)	(None, 16, 16, 256)	1024
max_pooling2d_3 (MaxPooling2D)	(None, 8, 8, 256)	0
dropout_3 (Dropout)	(None, 8, 8, 256)	0
flatten (Flatten)	(None, 16384)	0
dense (Dense)	(None, 512)	8389120
dropout_4 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 105)	53865

Total params: 8,833,321
Trainable params: 8,832,361
Non-trainable params: 960

Architecture du modèle entrainé sur la base de donnée Pins Face

Pour ce modèle, nous avons décidé de changer drastiquement d'architecture, et diminuer aussi le nombre de paramètre. Celui-ci possède 8 833 321

paramètres. Nous avons réussi à obtenir jusqu'à 78% de précision sur les données de validation et 91% sur les données d'entrainement.

Cependant, tous ces modèles sont sujet au surentrainement. Ainsi nous avons exploré un certain nombre de techniques supposés permettre le réduire.

b. Techniques utilisées afin de réduire le surentrainement

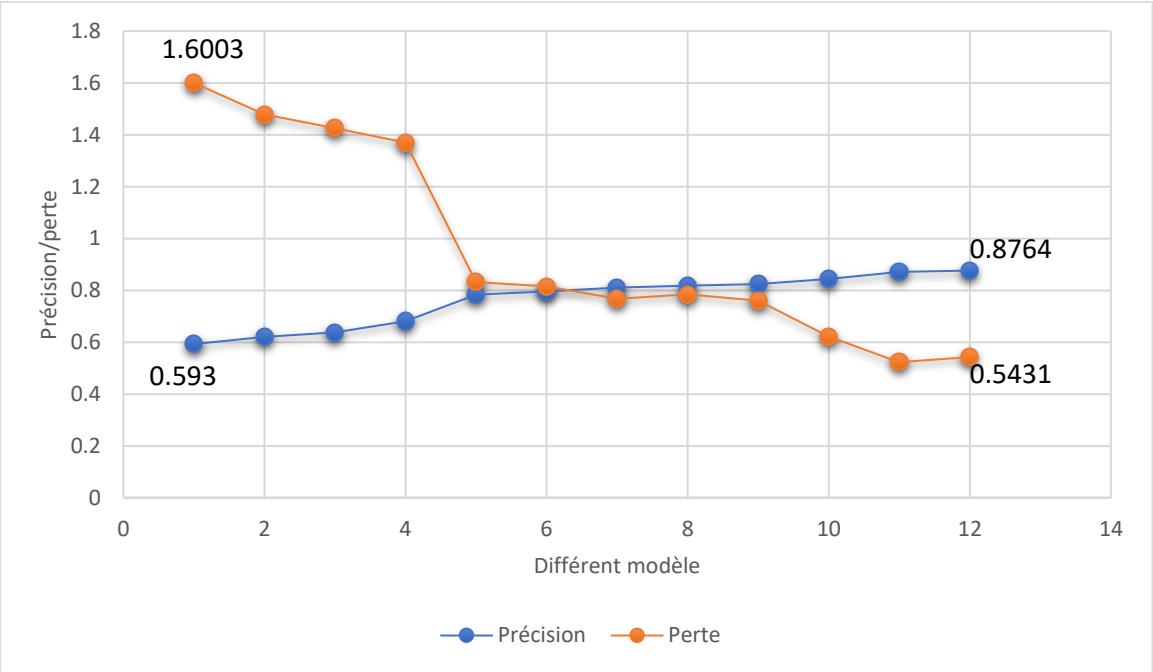
Dans notre recherche d'amélioration de la précision sur les données de test et la réduction du surentrainement, nous sommes venus à tester les paramètres suivants :

Paramètres	Observations
Techniques de régularisation L1, L2 : La régularisation L1, également appelée régression au lasso, ajoute la « valeur absolue de l'amplitude » du coefficient en tant que terme de pénalité à la fonction de perte. La régularisation L2, également appelée régression de crête, ajoute la "magnitude au carré" du coefficient comme terme de pénalité à la fonction de perte.	Nous n'avons pas observé de grandes améliorations sur les modèles, mais plutôt une diminution des performances.
Random_forest : Elle consiste à sélectionner des échantillons aléatoires à partir d'un ensemble de données. A partir de celles-ci l'algorithme construira un arbre de décision pour chaque donnée d'apprentissage. Elle procèdera ensuite en faisant la moyenne de l'arbre de décision et finalement sélectionnera le meilleur résultat de prédiction final.	Les améliorations étaient moindres, on a plutôt observé, ou bien des résultats identiques, ou légèrement inférieurs à la moyenne.
Early Stopping : C'est une technique qui consiste à arrêter l'entraînement du modèle dès que la performance sur les données	Mauvais résultats sur cette technique étant donné que le modèle n'arrivait pas à bien généraliser sur les données de test.

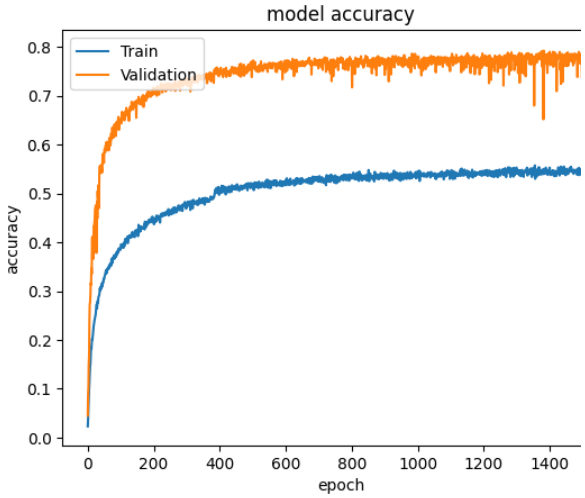
de validation commence à se détériorer.	
<u>Validation croisée :</u> La validation croisée est une méthode statistique d'évaluation et de comparaison des algorithmes d'apprentissage en divisant les données en deux segments : l'un utilisé pour apprendre ou former un modèle et l'autre utilisé pour valider le modèle.	Nous avons observé une légère augmentation de la précision, de l'ordre de 1 à 2% grâce à cette technique
<u>Callback : ReduceLROnPlateau</u> C'est est une technique de planification qui diminue le taux d'apprentissage lorsque la métrique spécifiée cesse de s'améliorer plus longtemps que le nombre de patience ne le permet. Ainsi, le taux d'apprentissage est maintenu le même, tant qu'il améliore la quantité métrique, mais le taux d'apprentissage est réduit lorsque les résultats stagnent.	L'une des techniques ayant le mieux fonctionnée. Elle a permis d'augmenter la précision sur les données de test de l'ordre de 2 à 7%. Elle est particulièrement efficace quand on observe une courbe oscillante. Elle lisse très bien le résultat, ce qui améliore l'apprentissage.
<u>Couche Dense :</u> Consiste à diminuer ou augmenter la complexité du modèle de manière à obtenir une meilleure généralisation sur les données de validation.	Résultats corrects mais assez complexes à paramétrer. Nous avons pu observer parfois une augmentation de la précision de près de 10%, ou sa diminution. La complexité était d'arriver à la bonne complexité, tout en permettant aux neurones d'apprendre facilement. Nous avons ainsi testé à la fois avec plusieurs couches Denses, mais aussi en augmentant simplement une seule couche Dense. Les résultats ont été meilleurs en utilisant une série de couches Dense, plutôt qu'en augmentant la complexité d'une seule.

c. Deuxième architecture :

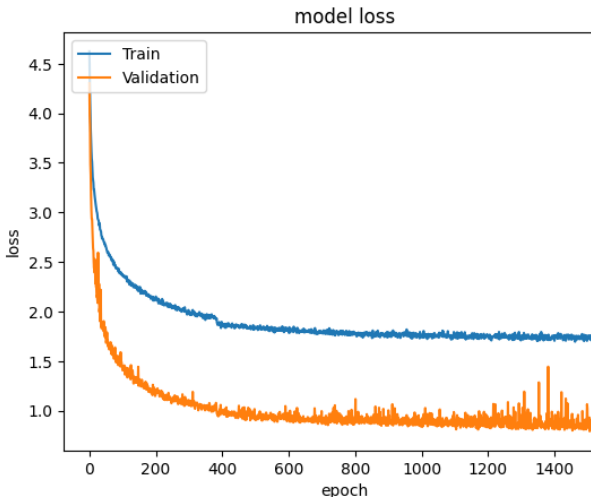
En parallèle de ce premier modèle, nous avons développé une autre architecture et l'avons entraînée de manière radicalement différente. Nous avons commencé avec peu de paramètres et avons augmenté progressivement leur nombre en cherchant à améliorer les performances. Nous avons utilisé la deuxième base de données (cf parties précédentes) pour entraîner ce modèle. À partir de ces essais successifs, nous avons tracé un graphique pour observer l'évolution des performances du modèle au fil du temps.



Pendant l'entraînement des différents modèles, nous avons été confrontés à une précision de validation supérieure à celle de l'entraînement.
Parallèlement, la perte était plus faible sur la validation que sur l'entraînement. Cette tendance est illustrée sur le graphique correspondant à l'entraînement du 5^{e} modèle.



Évolution de la précision pendant l'entrainement du 5e modèle



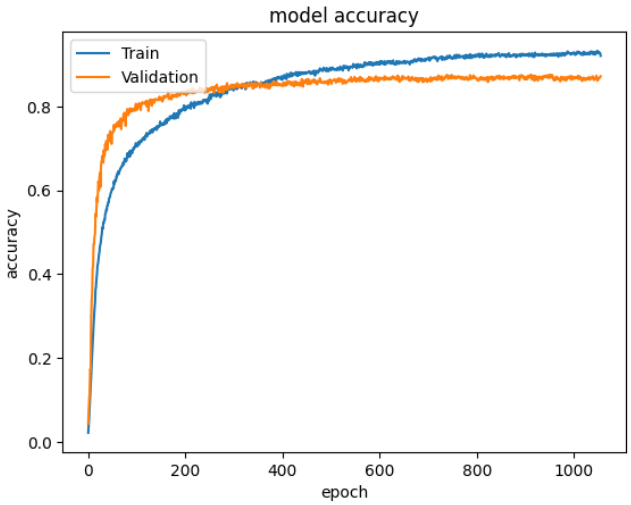
Évolution de la perte pendant l'entrainement du 5e modèle

D'après nos recherches, cette différence de performance entre l'entraînement et la validation pourrait potentiellement être due aux différentes couches de dropout. En effet, ces couches désactivent une partie des neurones uniquement pendant l'entraînement, tandis que tous les neurones sont utilisés lors de la validation. Cette différence pourrait également être due aux couches de normalisation, qui réduisent le biais des images. De plus, cette erreur pourrait provenir de la répartition des données, qui fournirait au réseau des données sur lesquelles il serait plus difficile de généraliser. Néanmoins, après avoir changé plusieurs fois les répartitions aléatoires en modifiant la graine de mélange, nous n'avons pas observé de changement. Cela nous amène à penser que ce phénomène ne vient pas du mélange des données.

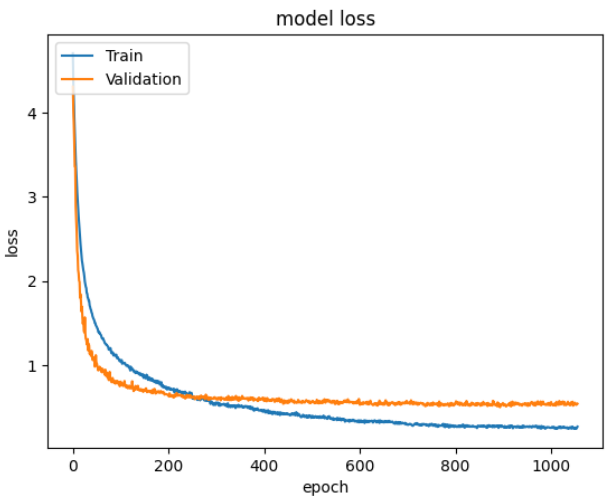
Pendant que nous cherchions à réduire cet écart entre les valeurs d'entraînement et de validation, nous avons continué à améliorer notre modèle. C'est ainsi que nous avons finalement obtenu un modèle ayant une précision de 0,8764 et une perte de 0,5431. L'architecture de ce modèle est la suivante :

Architecture du 12e modèle

Nous pouvons observer l'évolution de ce modèle avec les deux graphiques suivants :



Évolution de la précision

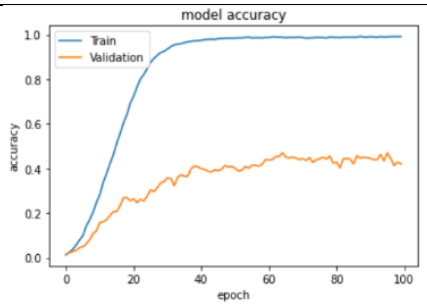
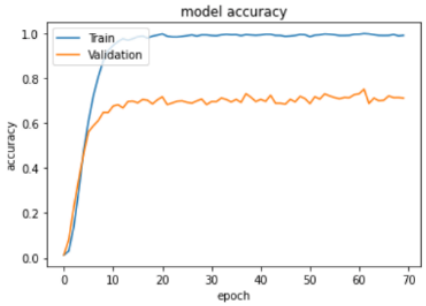


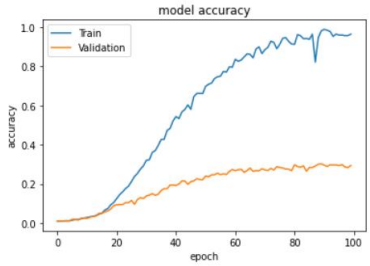
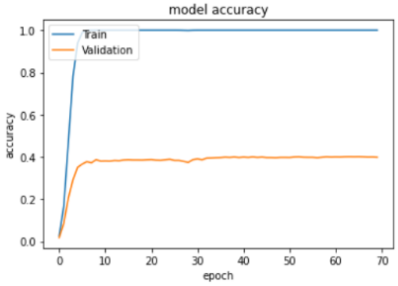
Évolution de la perte

Nous sommes convaincus qu'avec plus de temps, nous aurions pu atteindre une précision de 90% grâce à cette architecture. En effet, comme le montrent les graphiques ci-dessus, le modèle commençait à montrer des signes de surapprentissage. Les couches de BatchNormalization et de Dropout étaient les seules présentes pour atténuer ce phénomène. Nous pourrions donc envisager d'ajouter des couches de régularisation, telles que la couche L2, ainsi que de la data augmentation pour améliorer les performances du modèle.

d. Utilisation du Transfer Learning :

Après de nombreux essais avec nos premiers modèles et par manque de temps, nous avons essayé le transfert learning. Nous avons pris les modèles obtenant les meilleurs résultats en reconnaissance faciale et classification d'images. Ainsi, nous avons testé les modèles suivants : ResNet50, MobileNetV2, VGG16, VGG19 et InceptionV3. Nous avons obtenu les résultats ci-dessous.

Modèles :	Résultat obtenu
ResNet50 : Ce modèle a de meilleures performances de temps et de mémoire par rapport à VGGNet. Il a obtenu des performances remarquables pour reconnaitre l'identité de personnes masquées. De plus, il est adapté pour les réseaux ayant un grand nombre de classes différentes.	loss: 2.3108e-05 - accuracy: 1.0000 - val_loss: 2.0742 - val_accuracy: 0.5512 - Comme on peut le voir ci-dessus, les résultats sont décevants sur les données de validation avec 55% de précision. Cela pourrait s'expliquer due au faible nombre de classes de notre PinsFaceDataset.
MobileNetV2 : Il s'agit ici d'un CNN avec une taille de modèle plus petite, moins de paramètres d'apprentissage et de quantité de calcul, et convient aux appareils mobiles. Il tire pleinement parti de ses ressources de calcul et améliore au mieux la précision du modèle. C'est pourquoi nous l'avons choisi étant donné notre puissance de calcul limitée pour l'entrainement.	 Les résultats sont décevants, cela est notamment dû à une forte perte sur les données de test. On obtient 93% sur les données d'entrainement et 42% sur les données de test.
VGG16 : VGG est un réseau convolutif spécifique conçu pour la classification et la localisation et dans lequel les caractéristiques de l'image du visage sont extraites par le modèle de réseau neuronal à convolution VGGNet, puis les dimensions des caractéristiques extraites sont réduites par PCA, et enfin la reconnaissance faciale est effectuée par la méthode de classification SVM.	 Ces résultats sont jusqu'ici les meilleurs que l'on obtiendra en transfert learning sur cette

	base de données. On observe 73% de précision sur les données de test, et 99,4% sur les données d'entrainement
<u>VGG19 :</u> La différence entre VGG19 est que VGG19 à trois couches convolutionnelles supplémentaires. Les autres fonctionnalités telles que les couches de regroupement, les couches entièrement connectées et les canaux de classification sont les mêmes pour les deux réseaux. Ainsi, ce modèle apprend plus profondément que le VGG16.	 Les résultats sur celui-ci sont bien différents du VGG16, on observe une incapacité du modèle à généralisé. On peut expliquer cela par l'augmentation de près de 25% du nombre de paramètres du modèle.
<u>InceptionV3 :</u> Inception-v3 est un réseau de neurones convolutifs de 48 couches de profondeur, qui est une version du réseau déjà formé sur plus d'un million d'images de la base de données ImageNet. Il est particulièrement performant sur une grande quantité de classes.	 Les résultats sont décevants ici aussi, avec un très bon apprentissage sur les données d'entrainement à près de 99.5%, mais une incapacité à généraliser sur les données de validation ou l'on obtient une précision de 39.8%. Cela peut s'expliquer par son grand nombre de paramètres à près de 25 millions.

VI. Conclusion et perspectives

En conclusion de notre projet, nous avons dû établir un système de reconnaissance faciale capable de détecter puis de classifier le visage. Notre principal problématique a été l'utilisation du Deep Learning. Nous avons rencontré et surmonté plusieurs difficultés. Pour résoudre de manière optimale notre problématique, nous nous sommes basées sur deux approches distinctes afin d'étudier quelle serait la meilleure pour la classification. L'une correspondant à un réseau de neurones convolutif apprenant sur des images comprenant l'ensemble du visage et une autre approche basée sur des images contenant les points d'intérêt de ceux-ci. Nous avons également essayé d'autres méthodes comme celle du transfer learning afin d'explorer toutes les pistes. Nos modèles ont donné de bons résultats mais bien sûr ceux-ci sont toujours améliorables.

VII. Références bibliographiques

https://www.kaggle.com/datasets/prashantarorat/facial-key-point-data

https://www.kaggle.com/code/kerimelebiler/face-recognition-with-cnn/data

https://medium.com/sciforce/face-detection-explained-state-of-the-art-methods-and-best-tools-f730fca16294

https://www.sicara.fr/blog-technique/2019-09-26-face-detectors-dsfd-state-of-the-art-algorithms

https://fr.blog.businessdecision.com/tutoriel-deep-learning-le-reseau-neuronal-convolutif-cnn/